Babel routing protocol
Internet-Draft
Intended status: Informational
Expires: December 7, 2018

B. Stark
AT&T

June 5, 2018

Babel Information Model
draft-ietf-babel-information-model-03

Abstract

This Babel Information Model can be used to create data models under various data modeling regimes (e.g., YANG). It allows a Babel implementation (via a management protocol such as netconf) to report on its current state and may allow some limited configuration of protocol constants.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on December 7, 2018.

Table of Contents

1. Introduction

Babel is a loop-avoiding distance-vector routing protocol defined in draft-ietf-babel-rfc6126bis [rfc6126bis]. draft-babel-7298bis [BABEL-HMAC] defines a security mechanism that allows Babel messages to be cryptographically authenticated, and draft-babel-dtls [BABEL-DTLS] defines a security mechanism that allows Babel messages to encrypted. This document describes an information model for Babel (including implementations using one of these security mechanisms) that can be used to created management protocol data models (such as a netconf [RFC6241] YANG data model).

Due to the simplicity of the Babel protocol, most of the information model is focused on reporting status of the Babel protocol, and very little of that is considered mandatory to implement (conditional on a management protocol with Babel support being implemented). Some parameters may be configurable; however, it is up to the Babel implementation whether to allow any of these to be configured within its implementation. Where the implementation does not allow configuration of these parameters, it may still choose to expose them as read-only.

The Information Model is presented using a hierarchical structure. This does not preclude a data model based on this Information Model from using a referential or other structure.

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119] and updated by RFC 8174 [RFC8174] .

1.2. Notation

This document uses a programming language-like notation to define the properties of the objects of the information model. An optional property is enclosed by square brackets, [ ], and a list property is indicated by two numbers in angle brackets, <m..n>, where m indicates the minimal number of values, and n is the maximum. The symbol * for n means no upper bound.

The object definitions use base types that are defined as follows:

base64 An opaque array of bytes.

boolean A type representing a boolean value.

counter A non-negative integer that monotonically increases. Counters may have discontinuities and they are not expected to persist across restarts.

credentials An opaque type representing credentials needed by a cryptographic mechanism to secure communication. Data models must expand this opaque type as needed and required by the security protocols utilized.

datetime A type representing a date and time using the Gregorian calendar. The datetime format MUST conform to RFC 3339 [RFC3339].

int A type representing signed or unsigned integer numbers. This information model does not define a precision nor does it make a distinction between signed and unsigned number ranges.

ip-address A type representing an IP address. This type supports both IPv4 and IPv6 addresses.

string	A type representing a human-readable string consisting of a (possibly restricted) subset of Unicode and ISO/IEC 10646 [ISO.10646] characters.
uri	A type representing a Uniform Resource Identifier as defined in STD 66 [RFC3986].

2. Overview

The Information Model is hierarchically structured as follows:

```
information object
   includes implementation version, router id, this node seqno,
     enable flag parameters, supported security mechanisms
   constants object (exactly one per information object)
      includes UDP port and optional multicast group
        parameters
   interfaces object
      includes interface reference, Hello seqno and intervals,
        update interval, link type, external cost parameters
      neighbors object
         includes neighbor IP address, Hello history, cost
           parameters
      security object (per interface)
         includes enable flag, self credentials (credential
           object), trusted credentials (credential object)
   security object (common to all interfaces)
      includes enable flag, self credentials (credential
        object), trusted credentials (credential object)
   routes object
      includes route prefix, source router, reference to
        advertising neighbor, metric, sequence number, whether
        route is feasible, whether route is selected
```

Following is a list of the data elements that an implementation can choose to allow to be configurable:

- enable/disable babel
- Constant: UDP port
- Constant: IPv6 multicast group
- Interface: Link type
- Interface: External cost (must be configurable if implemented, but implementation is optional)

- Interface: enable/disable babel on this interface
- Interface: enable/disable message log
- Security: enable/disable this security mechanism
- Security: self credentials
- Security: trusted credentials
- Security: enable/disable security log

Note that this overview is intended simply to be informative and is not normative. If there is any discrepancy between this overview and the detailed information model definitions in subsequent sections, the error is in this overview.

3. The Information Model

3.1. Definition of babel-information-obj

```
  object {
       string                babel-implementation-version;
       boolean               babel-enable;
       base64                babel-self-router-id;
       string                babel-supported-link-types<1..*>;
      [int                   babel-self-seqno;]
       string                babel-metric-comp-algorithms<1..*>;
       string                babel-security-supported<0..*>;
       babel-constants-obj   babel-constants;
       babel-interfaces-obj  babel-interfaces<0..*>;
       babel-routes-obj      babel-routes<0..*>;
       babel-security-obj    babel-security<0..*>;
    }babel-information-obj;
```

babel-implementation-version: the version of this implementation of the Babel protocol

babel-enable: if true, the babel implementation is running; if false, the babel implementation is not currently running; MAY be configurable to allow babel to be started or stopped

babel-self-router-id: the router-id used by this instance of the Babel protocol to identify itself; draft-ietf-babel-rfc6126bis [rfc6126bis] describes this as an arbitrary string of 8 octets

babel-supported-link-types: set of values of supported link types where the following enumeration values MUST be supported when applicable: "ethernet", "wireless", "tunnel", and "other"

babel-self-seqno: the current sequence number included in route updates for routes originated by this node

babel-metric-comp-algorithms: a set of names of supported cost computation algorithms; possible values include "k-out-of-j", "ETX"

babel-security-supported: list of supported security mechanisms; as babel security mechanisms are defined, they will need to indicate what enumeration value is to be used to represent them in this parameter

babel-constants: a babel-constants-obj object

babel-interfaces: a set of babel-interface-obj objects

babel-security: a babel-security-obj object that applies to all interfaces; if this object is implemented, it allows a security mechanism to be enabled or disabled in a manner that applies to all Babel messages on all interfaces

babel-routes: a set of babel-route-obj objects; includes received and routes routes

3.2. Definition of babel-constants-obj

```
  object {
       int          babel-udp-port;
      [ip-address   babel-mcast-group-ipv6;]
   }babel-constants-obj;
```

babel-udp-port: UDP port for sending and listening for Babel messages; default is 6696; MAY be configurable

babel-mcast-group-ipv6: multicast group for sending and listening to multicast announcements on IPv6; default is ff02:0:0:0:0:0:1:6; MAY be configurable

3.3. Definition of babel-interfaces-obj

```
  object {
       string               babel-interface-reference;
      [boolean              babel-interface-enable;]
       int                  babel-link-type;
      [int                  babel-mcast-hello-seqno;]
      [int                  babel-ucast-hello-seqno;]
      [int                  babel-mcast-hello-interval;]
      [int                  babel-ucast-hello-interval;]
      [int                  babel-update-interval;]
      [int                  babel-external-cost;]
      [boolean              babel-message-log-enable;]
      [babel-log-obj        babel-message-log<0..*>;]
       babel-neighbors-obj  babel-neighbors<0..*>;
      [babel-security-obj   babel-interface-security<0..*>;]
   }babel-interfaces-obj;
```

babel-interface-reference: reference to an interface object as defined by the data model (e.g., YANG, BBF TR-181); data model is assumed to allow for referencing of interface objects which may be at any layer (physical, Ethernet MAC, IP, tunneled IP, etc.); referencing syntax will be specific to the data model; if there is no set of interface objects available, this should be a string that indicates the interface name used by the underlying operating system

babel-interface-enable: if true, babel sends and receives messages on this interface; if false, babel messages received on this interface are ignored and none are sent; MAY be configurable

babel-link-type: indicates the type of link; set of values of supported link types where the following enumeration values MUST be supported when applicable: "ethernet", "wireless", "tunnel", and "other"; additional values MAY be supported; MAY be configurable

babel-mcast-hello-seqno: the current sequence number in use for multicast hellos sent on this interface

babel-ucast-hello-seqno: the current sequence number in use for unicast hellos sent on this interface

babel-mcast-hello-interval: the current multicast hello interval in use for hellos sent on this interface

babel-ucast-hello-interval: the current unicast hello interval in use for hellos sent on this interface

babel-update-interval: the current update interval in use for this interface

babel-external-cost: external input to cost of link of this interface; if supported, this is a value that is added to the metrics of routes learned over this interface; how an implementation uses the value is up to the implementation, which means the use may not be consistent across implementations; MUST be configurable if implemented

babel-message-log-enable: if true, logging of babel messages received on this interface is enabled; if false, babel messages are not logged; MUST be configurable, if implemented

babel-message-log: log entries that have timestamp of a received Babel message and the entire received Babel message (including Ethernet frame and IP headers, if possible); an implementation must restrict the size of this log, but how and what size is implementation-specific

babel-neighbors: a set of babel-neighbors-obj objects

babel-interface-security: a babel-security-obj object that applies to this interface; if implemented, this allows security to be enabled only on specific interfaces or allows different security mechanisms to be enabled on different interfaces

3.4. Definition of babel-neighbors-obj

```
  object {
       ip-address           babel-neighbor-address;
      [string               babel-hello-mcast-history;]
      [string               babel-hello-ucast-history;]
       int                  babel-txcost;
       int                  babel-exp-mcast-hello-seqno;
       int                  babel-exp-ucast-hello-seqno;
       int                  babel-neighbor-ihu-interval;
      [int                  babel-rxcost]
      [int                  babel-cost]
   }babel-neighbors-obj;
```

babel-neighbor-address: (IPv4 or v6) address the neighbor sends messages from

babel-hello-mcast-history: the multicast Hello history of whether or not the multicast Hello messages prior to babel-exp-mcast-hello-seqno were received, with a "1" for the most recent Hello placed in the most significant bit and prior Hellos shifted right

(with "0" bits placed between prior Hellos and most recent Hello for any not-received Hellos); represented as a string using utf-8 encoded hex digits where a "1" bit = Hello received and a "0" bit = Hello not received; see draft-ietf-babel-rfc6126bis [rfc6126bis] section A.1

babel-hello-ucast-history: the unicast Hello history of whether or not the unicast Hello messages prior to babel-exp-ucast-hello-seqno were received, with a "1" for the most recent Hello placed in the most significant bit and prior Hellos shifted right (with "0" bits placed between prior Hellos and most recent Hello for any unreceived Hellos); represented as a string using utf-8 encoded hex digits where a "1" bit = Hello received and a "0" bit = Hello not received; see draft-ietf-babel-rfc6126bis [rfc6126bis] section A.1

babel-txcost: transmission cost value from the last IHU packet received from this neighbor, or maximum value (infinity) to indicates the IHU hold timer for this neighbor has expired

babel-exp-mcast-hello-seqno: expected multicast Hello sequence number of next Hello to be received from this neighbor; if multicast Hello messages are not expected, or processing of multicast messages is not enabled, this MUST be 0

babel-exp-ucast-hello-seqno: expected unicast Hello sequence number of next Hello to be received from this neighbor; if unicast Hello messages are not expected, or processing of unicast messages is not enabled, this MUST be 0

babel-neighbor-ihu-interval: current IHU interval for this neighbor

babel-rxcost: reception cost calculated for this neighbor; this value is usually derived from the Hello history, which may be combined with other data, such as statistics maintained by the link layer; the rxcost is sent to a neighbour in each IHU

babel-cost: link cost is computed from the values maintained in the neighbour table: the statistics kept in the neighbour table about the reception of Hellos, and the txcost computed from received IHU packets

3.5. Definition of babel-security-obj

```
  object {
       string                babel-security-mechanism
       boolean               babel-security-enable;
       babel-credential-obj  babel-security-self-cred<0..*>;
       babel-credential-obj  babel-security-trust<0..*>;
      [boolean               babel-credvalid-log-enable;]
      [babel-log-obj         babel-credvalid-log<0..*>;]
   }babel-security-obj;
```

babel-security-mechanism: the name of the security mechanism this object instance is about; the value MUST be the same as one of the enumerations listed in the babel-security-supported parameter

babel-security-enable: if true, the security mechanism is running; if false, the security mechanism is not currently running; MAY be configurable to allow security mechanism to be started or stopped

babel-security-self-cred: credentials this router presents to participate in the enabled security mechanism; any private key component of a credential MUST NOT be readable; adding and deleting credentials MAY be allowed

babel-security-trust: a set of babel-credential-obj objects that identify the credentials of routers whose babel messages may be trusted or of a certificate authority (CA) whose signing of a router's credentials implies the router credentials can be trusted, in the context of this security mechanism; how a security mechanism interacts with this list is determined by the mechanism; a security algorithm may do additional validation of credentials, such as checking validity dates or revocation lists, so presence in this list may not be sufficient to determine trust; adding and deleting credentials MAY be allowed

babel-credvalid-log-enable: if true, logging of messages that include credentials used for authentication is enabled; if false, these messages are not logged; MUST be configurable, if implemented

babel-credvalid-log: log entries that have the timestamp a message containing credentials used for peer authentication (e.g., DTLS Server Hello) was received on a Babel port, and the entire received message (including Ethernet frame and IP headers, if possible); an implementation must restrict the size of this log, but how and what size is implementation-specific

3.6. Definition of babel-routes-obj

```
  object {
       ip-address           babel-route-prefix;
       int                  babel-route-prefix-length;
       base64               babel-route-router-id;
       string               babel-route-neighbor;
      [int                  babel-route-received-metric;]
      [int                  babel-route-calculated-metric;]
       int                  babel-route-seqno;
       ip-address           babel-route-next-hop;
       boolean              babel-route-feasible;
       boolean              babel-route-selected;
   }babel-routes-obj;
```

babel-route-prefix: Prefix (expressed in IP address format) for which this route is advertised

babel-route-prefix-length: Length of the prefix for which this route is advertised

babel-route-router-id: router-id of the source router for which this route is advertised

babel-route-neighbor: reference to the babel-neighbors entry for the neighbor that advertised this route

babel-route-received-metric: the metric with which this route was advertised by the neighbor, or maximum value (infinity) to indicate a the route was recently retracted and is temporarily unreachable (see Section 3.5.5 of draft-ietf-babel-rfc6126bis [rfc6126bis]); this metric will be 0 (zero) if the route was not received from a neighbor but was generated through other means; either babel-route-calculated-metric or babel-route-received-metric MUST be provided

babel-route-calculated-metric: a calculated metric for this route; how the metric is calculated is implementation-specific; maximum value (infinity) indicates the route was recently retracted and is temporarily unreachable (see Section 3.5.5 of draft-ietf-babel-rfc6126bis [rfc6126bis]); either babel-route-calculated-metric or babel-route-received-metric MUST be provided

babel-route-seqno: the sequence number with which this route was advertised

babel-route-next-hop: the next-hop address of this route; this will be empty if this route has no next-hop address

babel-route-feasible: a boolean flag indicating whether this route is feasible, as defined in Section 3.5.1 of draft-ietf-babel-rfc6126bis [rfc6126bis])

babel-route-selected: a boolean flag indicating whether this route is selected, i.e., whether it is currently being used for forwarding and is being advertised

4. Common Objects

4.1. Definition of babel-credential-obj

```
  object {
       credentials          babel-cred;
 }babel-credential-obj;
```

babel-cred: a credential, such as an X.509 certificate, a public key, etc. used for signing and/or encrypting babel messages

4.2. Definition of babel-log-obj

```
  object {
       datetime           babel-log-time;
       string             babel-log-entry;
 }babel-log-obj;
```

babel-log-time: the date and time (according to the device internal clock setting, which may be a time relative to boot time, acquired from NTP, configured by the user, etc.) when this log entry was created

babel-log-entry: the logged message, as a string of utf-8 encoded hex characters

5. Extending the Information Model

Implementations MAY extend this information model with other parameters or objects. For example, an implementation MAY choose to expose babel route filtering rules by adding a route filtering object with parameters appropriate to how route filtering is done in that implementation. The precise means used to extend the information model would be specific to the data model the implementation uses to expose this information.

6. Security Considerations

This document defines a set of information model objects and parameters that may be exposed to be visible from other devices, and some of which may be configured. Any mechanism or protocol that is used to transmit this information or allow for its configuration is also responsible for ensuring this is done so in a secure manner.

This information model defines objects that can allow credentials (for this device, for trusted devices, and for trusted certificate authorities) to be added and deleted. Public keys and shared secrets may be exposed through this model. This model requires that private keys never be exposed. The Babel security mechanisms that make use of these credentials are not defined or identified in this model.

7. IANA Considerations

This document makes no IANA requests.

8. Acknowledgements

Juliusz Chroboczek, Toke Hoeiland-Joergensen, and David Schinazi have been very helpful in refining this information model.

The language in the Notation section was mostly taken from RFC 8193 [RFC8193].

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[rfc6126bis] Chroboczek, J., "The Babel Routing Protocol", Work in Progress, draft-ietf-babel-rfc6126bis, October 2017.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

9.2. Informative References

[BABEL-DTLS]
Schinazi, D., "TBD", Work in Progress, rfc6347, March 2018.

[BABEL-HMAC]
Ovsienko, D., "Babel HMAC Cryptographic Authentication", Work in Progress, draft-ovsienko-babel-rfc7298bis, March 2018.

[ISO.10646]
International Organization for Standardization, "Information Technology - Universal Multiple-Octet Coded Character Set (UCS)", ISO Standard 10646:2014, 2014.

[RFC3339] Klyne, G. and C. Newman, "Date and Time on the Internet: Timestamps", RFC 3339, DOI 10.17487/RFC3339, July 2002, <https://www.rfc-editor.org/info/rfc3339>.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <https://www.rfc-editor.org/info/rfc3986>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

[RFC8193] Burbridge, T., Eardley, P., Bagnulo, M., and J. Schoenwaelder, "Information Model for Large-Scale Measurement Platforms (LMAPs)", RFC 8193, DOI 10.17487/RFC8193, August 2017, <https://www.rfc-editor.org/info/rfc8193>.

Appendix A. Open Issues

This draft must be reviewed against draft-ietf-babel-rfc6126bis. [I feel like this has been adequately done, but I could be wrong.]

Following are some issues where a conscious decision may be useful:

1. babel-interfaces-obj: Juliusz:"This needs further discussion, I fear some of these are implementation details." [In the absence of discussion, the current model stands. Note that all but link-type and the neighbors sub-object are optional; if an

implementation does not have any of the optional elements then it simply doesn't have them and that's fine.]

2. Would it be useful to define some parameters for reporting statistics or logs? [2 logs are now included. If others are needed they need to be proposed.]

3. Would it be useful to define some parameters specifically for security anomalies? [The 2 logs should be useful in identifying security anomalies. If more is needed, someone needs to propose.]

4. I created a basic security model. It's useful for single (or no) active security mechanism (e.g., just HMAC, just DTLS, or neither); but not multiple active (both HMAC and DTLS -- which is not the same as HMAC of DTLS and would just mean that HMAC would be used on all unencrypted messages -- but right now the model doesn't allow for configuring HMAC of unencrypted messages for routers without DTLS, while DTLS is used if possible). OK?

5. babel-external-cost may need more work. [if no comment, it will be left as is]

6. babel-hello-[mu]cast-history: the Hello history is formated as 16 bits, per A.1 of 6126bis. Is that a too implementation specific? [We also now have an optional-to-implement log of received messages, and I made these optional. So maybe this is ok?]

7. rxcost, txcost, cost: is it ok to model as integers, since 6126bis 2.1 says costs and metrics need not be integers. [I have them as integers unless someone insists on something else.]

8. Should babel link types have an IANA registry? [Right now, none is defined.]

9. For the security log, should it also log whether the credentials were considered ok? [Right now it doesn't and I think that's ok because if you log Hellos it was ok and if you don't it wasn't.]

Closed Issues:

Closed by defining base64 type and using it for all router IDs: "babel-self-router-id: Should this be an opaque 64-bit value instead of int?"

Closed as "No": Do we need a registry for the supported security mechanisms? [Given the current limited set, and unlikelihood of

massive expansion, I don't think so. But we can if someone wants it.]

Appendix B. Change Log

Individual Drafts:

v00 2016-07-07 EBD Initial individual draft version

v01 2017-03-13 Addressed comments received in 2016-07-15 email from J. Chroboczek

Working group drafts:

v00 2017-07-03 Addressed points noted with "oops" in https://www.ietf.org/proceedings/98/slides/slides-98-babel-babel-information-model-00.pdf

v01 2018-01-02 Removed item from issue list that was agreed (in Prague) not to be an issue. Added description of data types under Notation section, and used these in all data types. Added babel-security and babel-trust.

v02 2018-04-05

- changed babel-version description to babel-implementation-version

- replace optional babel-interface-seqno with optional babel-mcast-hello-seqno and babel-ucast-hello-seqno

- replace optional babel-interface-hello-interval with optional babel-mcast-hello-interval and babel-ucast-hello-interval

- remove babel-request-trigger-ack

- remove "babel-router-id: router-id of the neighbor"; note that parameter had previously been removed but description had accidentally not been removed

- added an optional "babel-cost" field to babel-neighbors object, since the spec does not define how exactly the cost is computed from rxcost/txcost

- deleted babel-source-garbage-collection-time

- change babel-lossy-link to babel-link-type and make this an enumeration; added at top level babel-supported-link-types so which are supported by this implementation can be reported

- changes to babel-security-obj to allow self credentials to be one or more instances of a credential object; allowed trusted credentials to include CA credentials; made some parameter name changes

- updated references and Introduction

- added Overview section

- deleted babel-sources-obj

- added feasible Boolean to routes

- added section to briefly describe extending the information model.

- deleted babel-route-neighbor

- tried to make definition of babel-interface-reference clearer

- added security and message logs

v03 2018-05-31

- added reference to RFC 8174 (update to RFC 2119 on key words)

- applied edits to Introduction text per Juliusz email of 2018-04-06

- Deleted sentence in definition of "int" data type that said it was also used for enumerations. Changed all enumerations to strings. The only enumerations were for link types, which are now "ethernet", "wireless", "tunnel", and "other".

- deleted [ip-address babel-mcast-group-ipv4;]

- babel-external-cost description changed

- babel-security-self-cred: Added "any private key component of a credential MUST NOT be readable;"

- hello-history parameters put recent Hello in most significant bit and length of parameter is not constrained.

- babel-hello-seqno in neighbors-obj changed to babel-exp-mcast-hello-seqno and babel-exp-ucast-hello-seqno

- added babel-route-neighbor back again; it was mistakenly deleted

- changed babel-route-metric and babel-route-announced-metric to babel-route-received-metric and babel-route-calculated-metric

- changed model of security object to put list of supported mechanisms at top level and separate security object per mechanism; this caused some other changes to the security object

Author's Address

Barbara Stark
AT&T
Atlanta, GA
US

Email: barbara.stark@att.com